

### REMARKS

The application includes claims 1-15, 17-23, and 25-28 prior to entering this amendment.

The examiner rejects claims 3-4, 11-13, and 25-28 under 35 U.S.C. § 112, first paragraph, as failing to comply with the enablement requirement.

The examiner rejects claims 1-15, 17-23, and 25-28 under 35 U.S.C. § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter that the applicants regard as their invention.

The examiner rejects claims 1, 2, 5-9, 11, and 12 under 35 U.S.C. § 102(e) as being anticipated by Arora et al. (U.S. Patent 6,625,693).

The examiner rejects claims 1-4 under 35 U.S.C. § 102(e) as being anticipated by Crump et al. (U.S. Patent 5,557,759).

The applicant amends claims 1-4, 10-15, 22, 25, and 28. The application remains with claims 1-15, 17-23, and 26-28 after this amendment.

The applicants add no new matter and request reconsideration.

### Drawing Amendments

The applicants submit a redlined Figure 1 to correct an error in the directional arrow between the instruction fetch unit 20 and the instruction cache 22. The amendments are fully supported in the specification as we explain in more detail in the immediately subsequent section on claim rejections under § 112.

### Claim Rejections Under § 112

The examiner rejects claims 3-4, 11-13, and 25-28 as failing to adequately enable how to retire back to cache executed instructions or executed micro-instructions. And the examiner rejects claims 1-15, 17-23 and 25-28 as being indefinite. The applicants traverse the rejections for the reasons that follow.

The specification and claims of the present application, as well as the parent application now U.S. patent number 6,662,297, adequately enable both the retiring of executed micro-ops and the inserting of micro-ops corresponding to interrupt servicing instructions. As detailed in Figure 1, an interrupt concentrator 14 detects a plurality of interrupt servicing instructions

AMENDMENT

PAGE 11 OF 14

DOC. NO. 5038-331  
SERIAL NO. 10/698,144

12a...12n and ranks them in accordance with a predetermined priority scheme. The interrupt concentrator 14 signals a core processor 16 of a detected interrupt servicing instruction. The core processor 16 includes an instruction queue mechanism 20, p-code decoder 18, and dispatch and execute unit 36. The instructions queue mechanism 20 includes an instruction fetch unit 20 that fetches instructions from memory and stores them in an instruction cache 22. Specification, page 5, lines 12-19. The instruction fetch unit 20 reads a predefined number of bytes, e.g., 32 bytes, from the instruction cache 22 and routes some lesser number of bytes, e.g., sixteen bytes, to the p-code decoder 18. The instruction fetch unit 20, therefore, provides instructions to and fetches instructions from the instruction cache 22, hence the need to change the directional arrow in Figure 1.

The p-code decoder 18 decodes the instructions received from the instruction fetch unit 20 into a set of typically multiple, reduced instructions, called micro-ops. Specification, page 3, line 30 to page 4, line 9.

The p-code decoder 18 routes micro-ops to a register alias table (RAT) (not shown in Figure 1). In the RAT, any register references and status information implicit within the micro-ops are incorporated before the micro-ops are stored in an instruction pool/reorder buffer (ROB) (not shown in Figure 1). Three parallel decoders included in the p-code decoder 18 have different capabilities to decode instructions into micro-ops. A microcode instruction sequencer included in the p-code decoder 18 decodes complex instructions, e.g., macro-ops, in accordance with a complex instruction set (CISC) processor architecture into reduced instructions, e.g. micro-ops, in accordance with a reduced instruction set (RISC) processor architecture. Specification, page 6, lines 1-9.

An "out-of-order execution unit 24 schedules and executes the micro-ops stored in the ROB that is loaded by p-code decoder 18. A reservation station (RS) within out-of-order execution unit 24 continuously scans the ROB for  $\mu$ OPs that are ready to be executed and dispatches them to one or more of five execution ports. The five execution ports preferably include a first port including an integer unit, a floating point unit, an address generation unit, a packed arithmetic logic unit (ALU) and a packed multiply unit. A second port includes an integer unit, a packed ALU and a packed shift unit. The third through fifth ports include a load unit, a store address calculation unit and a store data unit." Specification, page 6, lines 10-18.

“Instruction retirement unit 26 simply stores the executed instructions from OOO execution unit 24, in their original consecutive order, into instruction cache 22 for future re-use by instruction fetch unit 20. Those of skill in the art will appreciate that instruction retirement ensures that instruction cache 22 thus will contain instructions that may be required by instruction fetch unit 20 when instruction fetch unit 20 performs its next instruction pre-fetch. This avoids the latencies related to memory bus contention and read access that otherwise would slow processing if the same instruction sequence is required to be performed again, as may often be the case.” Specification, page 6, lines 19-26.

The indicated passages as well as the detailed description of the interrupt concentrator 14 shown in Figure 2, the state machines shown in Figures 3A-3B, and the flowchart shown in Figure 4 should obviate the examiner’s rejections based on § 112, first paragraph.

With regard to the examiner’s claim rejections under § 112, second paragraph, the applicants amend the claims to particularly point out and distinctly claim the subject matter claimed. The applicants believe the amended drawings and noted passages should make clear to the examiner, for example, that the fetch unit 20 may fetch micro-ops for re-execution.

The applicants amend claims 14, 22, 25, and 28 to obviate the examiner’s rejections.

Independent claims 10, 13, 14, 22, 25, and 28 and their corresponding dependent claims 15-21, 23, and 26-27 are in condition for the examiner’s allowance.

### Claim Rejections Under § 102

The examiner alleges claims 1-2, 5-9, and 11-12 are old over Arora. And the examiner alleges claims 1-4 are old over Crump. The applicants traverse the examiner’s rejections for the reasons that follow.

Claim 1 recites *concurrently in-line staging the interrupt servicing instructions inserted into mainline program instructions within the instruction queue mechanism resulting in allocating core processor bandwidth between the interrupt servicing and mainline program instructions*. The examiner maintains that Arora handles an exception upon an interrupt raised by the exception, and the interrupt is detected to handle the exception. According to the examiner, the exception handlers are inserted into a pool or queue. But the claim recites *concurrently in-line staging the interrupt servicing instructions inserted into the mainline program instructions*, and not inserting the lightweight handlers into a pool or queue as asserted

by the examiner. Arora teaches no such concurrent in-line staging of interrupt instructions inserted into mainline program instructions.

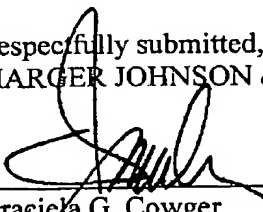
Crump, for its part, discloses an interrupt servicing system in which an interrupt vector request register receives and retains the appropriate *vector address* when an interrupt occurs. The mainline program code continues to execute while the cache determines a "hit or miss status for the interrupt vector and fetches the code on a miss." Column 18, lines 24-25. "Only after the interrupt service routine code is present in the cache will the vector occur, whereupon *the mainline program counter will be stored in the BALI and IVRR will be loaded into the program counter*. This allows useful work to be performed while the interrupt is pending." Column 18, lines 25-29. It doesn't appear to the applicants that Crump discloses the cited limitation, particularly where Crump stores its program counter so that it knows where to return upon an interrupt. If Crump were *concurrently in-line staging* interrupt servicing instructions inserted into the mainline program instructions as recited, it would have no need for an instruction cache that "time-shared between code fetches, interrupt requests, and line bus snooping." Column 18, lines 30-42.

### Conclusion

For the foregoing reasons, the applicants request reconsideration and allowance of all remaining claims. The applicants encourage the examiner to telephone the undersigned at (503) 222-3613 if it appears that an interview would be helpful in advancing the case.

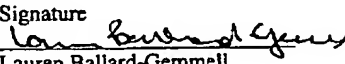
Customer No. 32231

Respectfully submitted,  
MARGER JOHNSON & McCOLLOM, P.C.

  
Graciela G. Cowger  
Reg. No. 42,444

MARGER JOHNSON & McCOLLOM, P.C.  
210 SW Morrison Street, Suite 400  
Portland, OR 97204  
503-222-3613

I hereby certify that this correspondence is being transmitted to the U.S. Patent and Trademark Office via facsimile number 571-273-8300, on July 7, 2006.

Signature   
Lauren Ballard-Gemmell

AMENDMENT

PAGE 14 OF 14

DOC. No. 5038-331  
SERIAL No. 10/698,144

Douglas D. Boom and Matthew M. Gilbert  
ALLOCATION OF PROCESSOR BANDWIDTH BY INSERTING INTERRUPT SERVICING INSTRUCTIONS TO INTERVENE MAIN  
PROGRAM IN INSTRUCTION QUEUE MECHANISM  
Attorney Docket No. 5038-331/Application No. 10/698,144

Annotated Sheet Showing Changes

FIG. 1

